Classifying and Identifying Servers for Biomedical Information Retrieval

Timothy B. Patrick, Ph.D.[1] and Gordon K. Springer, Ph.D.[2]
[1]Medical Informatics Group, University of Missouri-Columbia
[2]Department of Computer Science, University of Missouri-Columbia

Useful retrieval of biomedical information from network information sources requires methods for organized access to those information sources. This access must be organized in terms of the information content of information sources and in terms of the discovery of the network location of those information sources. We have developed an approach to providing organized access to information sources based on a scheme of hierarchical classifiers and identifiers of the servers providing access to those information sources. This approach uses MeSH tree numbers as both classifiers and identifiers of servers. MeSH tree numbers are used to indicate the information content of servers, and also as OSF/DCE server identifiers. This allows the identity and location of a server providing access to a given information source to be determined from the information classification of that information source.

A basic goal of the Medical Informatics community is the development of distributed information systems that will facilitate truly useful retrieval of biomedical information from large distributed information spaces [1],[2]. Such systems should be based on sufficiently general paradigms of information retrieval. They must, for example, treat retrieval of information from document information sources (e.g., Medline) equally with retrieval of information from computational information sources (e.g., molecular sequence analysis tools.) Such systems must also facilitate the use of combinations of information sources [3]. And above all, such systems must have a sound logical foundation that allows for systematic, organized access to information sources wherever, and whenever those information sources are available [4]. Systematic and organized access to information sources requires two types of organization. First, access to information sources must be organized in terms of the information content of information sources. Second, access to information sources must be organized in terms of the discovery of the network location of information sources.

Several very useful tools for browsing the Internet are currently available, most notably Gopher and Mosaic/World Wide Web [5]. These tools support information retrieval from large distributed information spaces. Using user menus and hyper-links, these tools support information retrieval that is organized in terms of the information content of information sources. In addition, these tools may be used to retrieve information equally from both document information sources and computational information sources. However, as currently utilized, these tools suffer from two defects. First, in order to access a specific information source, a user may be required to follow a circuitous and time-consuming path from menu to menu or hyper-link to hyper-link [6]. Second, rather than determining the network location of a server providing access to a desired information source at the time access is requested, these tools *hardcode* the network location of a server in the identifier of an information source [7]. A disadvantage of this approach is that if the network location of the server has changed, and the identifier of the information source has not been updated, an attempt to access the information source will fail.

We have developed an approach to information source access which uses standard MeSH tree numbers [8] as both hierarchical classifiers and hierarchical identifiers of the servers providing access to information sources. This approach supports organized access to information sources both in terms of their information content and in terms of the discovery of their network location. The identity of a server providing access to an information source is determined directly from the information classification of the information source. The network location of the server is determined at the time access to the information source is requested.

OVERVIEW

Our approach provides support for information retrieval from large distributed information spaces. The information space is assumed to consist of multiple possibly overlapping information domains. An information space is represented by one or more hierarchical information vocabulary trees. Information domains are represented by subtrees of the information space. We use the MeSH vocabulary trees to represent the biomedical information space and MeSH subtrees to represent particular biomedical information domains. MeSH codes are used to classify the information content of both information sources and servers

0195-4210/94/$5.00

providing access to information sources. The appropriate server class for a desired information source is determined from the MeSH codes used to classify the information source. The MeSH code used to classify the server is transformed to provide a server identifier. A *name service* is used to lookup the current network locations associated with the server identifier. The server instances at the reported network locations are queried for current support of the desired information source. When a positive response is received, a request to access the information source is sent to the corresponding server instance.

IMPLEMENTATION

We have implemented this approach in a client/server, distributed information system, *MUinfo* [9]. *MUinfo* is based on the Open Software Foundation/Distributed Computing Environment (OSF/DCE) [10],[11]. *MUinfo* is currently in testing with servers running on DEC Alpha and IBM RS/6000 workstations at the University of Missouri-Columbia and on an RS/6000 and the Cray Y-MP C90 at the Pittsburgh Supercomputing Center.

MUinfo

The main components of *MUinfo* are (1) an information sources database (ISDB) the entries of which describe application programs, (2) a set of utilities for processing the entries in the ISDB and for combining information sources, (3) a standard type of client and server which provide command string access to information sources, where an information source is an application program, (4) a set of intermediary servers which provide gateways to environments other than OSF/DCE, and (5) a Motif user interface. The ISDB is similar to the National Library of Medicine Information Sources Map [2], except that the ISDB entries describe application programs rather than databases. The ISDB entries include sufficient information to build command strings to access the application programs. The ISDB utilities provide a means for constructing command strings. The ISDB utilities also support the use of combinations of information sources by allowing data to be filtered from the results returned by one information source so that it may be used as input to another information source. The standard type client and server allow a constructed command string to be sent to an appropriate server in order to execute a desired application program. Such an application program may itself be a client which accesses a remote server. One example of this is a telnet script that we use to retrieve gene map information from Online Mendelian Inheritance in Man (OMIM.) The intermediary servers provide a means to access information sources that are not directly accessible by means of OSF/DCE. An example *OSF/DCE to FTP* intermediary server is described by [12].

OSF/DCE Server Identifiers

An OSF/DCE server is identified by a standard form Object Universal Unique Identifier (Object UUID) and a standard form Interface Universal Unique Identifier (Interface UUID). A UUID consists of 32 hexadecimal digits. An Object UUID and Interface UUID pair may be used to uniquely identify a server. The following is an example Object UUID and Interface UUID pair.

Object 03746500-0000-0000-dddd-08002b37598d
Interface 10000000-0000-0000-dddd-08002b37598d

Using an Object UUID and Interface UUID pair, OSF/DCE servers register themselves with the OSF/DCE *name service*. The OSF/DCE name service may be used to look up the network locations currently associated with a particular Object UUID and Interface UUID pair. A helpful discussion of the OSF/DCE name service may be found in [11].

Classification and Identification of Servers

In *MUinfo* we use the Object UUID to encode the *information domain* supported by a server and we use the Interface UUID to encode the *server type*.

Information domains are represented by MeSH subtrees. A server is classified by the MeSH tree number that is the root of the server's information domain. The Object UUID of a server is derived from the MeSH tree number used to classify the server. For example, Figure 1 depicts the derivation of the server Object UUID

Object 03746500-0000-0000-dddd-08002b37598d

A server with this Object UUID is associated with the information domain represented by the MeSH subtree

Mouth Diseases (C7.465)
Periodontal Diseases (C7.465.714)
Alveolar Bone Loss (C7.465.714.150)

A server providing access to information sources in this domain is classified with the root of the domain, i.e., *Mouth Diseases (C7.465)*. The Object UUID assigned to a server providing access to information sources in this domain is derived from the MeSH tree

number *C7.465*. As shown in Figure 1, the initial portion of the Object UUID, "03746500-0000-0000", is derived from the MeSH tree number for the MeSH heading "Mouth Diseases"-- C7.465 -- with the "C" mapped to its alphabetic position, i.e., "03". The latter portion of the Object UUID, "dddd-08002b37598d", is a common UUID stub.

(1) The information domain *Mouth Diseases* is represented by the MeSH subtree

Mouth Diseases (C7.465)
Periodontal Diseases
Alveolar Bone Loss

(2) C7.465 is mapped to the UUID fragment

03746500-0000-0000

(3) The UUID fragment is concatenated with the UUID stub.

03746500-0000-0000 + dddd-08002b37598d

(4) The resulting Object UUID is

03746500-0000-0000-dddd-08002b37598d

Figure 1 Derivation of an Object UUID

Different types of servers may be associated with the same information domain. A server's Interface UUID encodes its *type* and implies the protocol that must be used to make requests of the server. For example, the protocol that must be used to make requests of a standard type *MUinfo* command string server is a particular set of OSF/DCE Remote Procedure Calls (OSF/DCE RPC.) We adopted the Interface UUID

Interface 10000000-0000-0000-dddd-08002b37598d

to represent that server type. Thus, the Object UUID and Interface UUID pair

Object 03746500-0000-0000-dddd-08002b37598d
Interface 10000000-0000-0000-dddd-08002b37598d

identifies a server that is of the *MUinfo* OSF/DCE RPC command string server type and that provides access to information sources in the information domain represented by the MeSH subtree

Mouth Diseases (C7.465)
Periodontal Diseases (C7.465.714)
Alveolar Bone Loss (C7.465.714.150)

Servers which support that same information domain but which require a different access protocol (e.g., a different set of remote procedure calls) have the same Object UUID but a different Interface UUID. In principle, even servers which require an access protocol other than OSF/DCE RPC (e.g., a Gopher server or a Mosaic/World Wide Web server) could register themselves with the OSF/DCE name service. The OSF/DCE name service could be used to lookup the network locations currently associated with such servers. The protocol implied by the server type could then be used to make requests of the server. In this way, an Object UUID and Interface UUID pair is analogous to an *unbound* URL (Uniform Resource Locator) [7]. An Object UUID and Interface UUID pair is an *unbound* URL in the sense that the network locations associated with it are determined at the time access to the corresponding server is required. Part of our current development effort is directed to building a Mosaic/World Wide Web server which is able to register itself with the OSF/DCE name service.

Organized Access to Information Sources

The *MUinfo* ISDB entries index information sources (i.e., application programs) with MeSH codes. Such indexing *locates* an information source in an information domain. The location of an information source in an information domain is used to determine the Object UUID of a server that will provide access to the information source. For example, consider an information source indexed with

Alveolar Bone Loss (C7.465.714.150)

The information source is located in the information domain represented by the MeSH subtree

Mouth Diseases (C7.465)
Periodontal Diseases (C7.465.714)
Alveolar Bone Loss (C7.465.714.150)

The root of that MeSH subtree is *Mouth Diseases (C7.465)*. A server classified with *Mouth Diseases (C7.465)* provides access to information sources located in that information domain. The Object UUID of such a server is

Object 03746500-0000-0000-dddd-08002b37598d

The initial portion of this Object UUID is constructed from the MeSH tree number "C7.465", with the "C"

mapped to its alphabetic position. Given this Object UUID, and given the Interface UUID for the type of server desired, the OSF/DCE name service is used to determine the current network locations associated with that Object UUID and Interface UUID pair. The server instances at each of the reported network locations are queried for current support of the *Alveolar Bone Loss* information source. Given a positive response, a request to access the information source is sent to the corresponding server instance. In the case of the standard *MUinfo* servers, that request is a command string required to execute the *Alveolar Bone Loss* information source.

Server Hierarchies

The approach we have developed allows us to use MeSH trees to represent not only information domains, but also logical hierarchies of types of servers, where the type of server is represented by a particular Interface UUID. Each node in a MeSH tree may represent a server Object UUID. Different levels in the tree may represent different levels of server responsibility. For example, Figure 2 depicts a server hierarchy corresponding to the MeSH subtree

Mouth Diseases (C7.465)
Periodontal Diseases (C7.465.714)
Alveolar Bone Loss (C7.465.714.150)

Subject to the length constraint on UUID's, a server Object UUID is constructed from the MeSH tree number of each non-terminal node in the subtree (e.g., *Mouth Diseases* and *Periodontal Diseases*.) These UUIDs are constructed in the usual way (see Figure 1.)

Organized Searching for Information Sources

Our representation of logical hierarchies of types of servers provides a basis for organized searching for information sources. The search is organized by the information classifications of information sources. For example, suppose we are searching for an information source to provide information in the category *Alveolar Bone Loss*. Furthermore, suppose that we intend to access that information source using an *MUinfo* OSF/DCE RPC command string server, i.e., a server with Interface UUID

Interface 10000000-0000-0000-dddd-08002b37598d

Using the server hierarchy depicted in Figure 2, we could first use the DCE name service to search for a server providing access to information sources in the information domain *Periodontal Diseases*. We could search for a server with Object UUID and Interface UUID

Object 03746571-4000-0000-dddd-08002b37598d
Interface 10000000-0000-0000-dddd-08002b37598d

If a server instance were available, it could be queried for an *Alveolar Bone Loss* information source. If such a server were not available, or did not provide access to an *Alveolar Bone Loss* information source, then the next server class in the hierarchy could be processed. That is, the DCE name service could be searched for a server providing access to information sources in the information domain *Mouth Diseases*. We could search for a server with Object UUID and Interface UUID

Object 03746500-0000-0000-dddd-08002b37598d
Interface 10000000-0000-0000-dddd-08002b37598d

If such a server were available, it could be queried for an *Alveolar Bone Loss* information source.

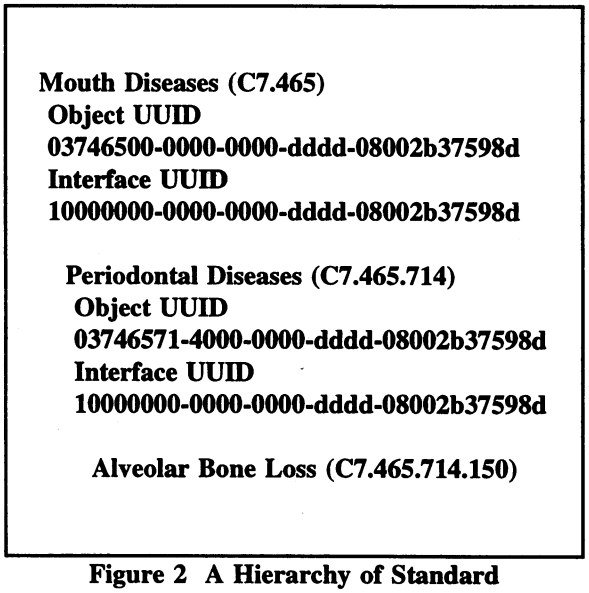

Figure 2 A Hierarchy of Standard *MUinfo* Servers

In the case of the standard *MUinfo* servers, such a query takes the form of a keyword search of an ISDB maintained at the server. For example, a query for an *Alveolar Bone Loss* information source takes the form of an ISDB search using the keyword *Alveolar Bone Loss*. If a matching entry is found in the server's ISDB, the ISDB entry is returned for processing by the client's ISDB utilities. A command string for the information source is constructed. The

information source is accessed by sending the command string back to the server.

SUMMARY AND CONCLUSION

We have developed an approach to providing organized access to information sources both in terms of their information content and in terms of the discovery of their network locations. Our current implementation of this approach uses standard MeSH tree numbers as hierarchical classifiers and also as hierarchical identifiers of the servers providing access to information sources. This approach allows the identity and location of a server providing access to an information source to be determined from the information classification of that information source. We hypothesize that this approach may be extended to standard public vocabularies other than MeSH, and to information domains other than biomedical information domains.

Our approach allows the use of MeSH trees to represent logical hierarchies of servers. These hierarchies of servers may be systematically searched for a server providing access to an information source in a desired category.

Although the use of MeSH trees to represent logical hierarchies of servers appears promising, serious questions remain about the best way to understand those hierarchies and the relationship between the classification of servers and the classification of information sources. In particular, the extent of the responsibilities, or the scope of the content, of any given server must be determined. We must ask, for example, whether a server identified by a given node in a MeSH tree is responsible for providing access to all information sources classified by descendants of that node. In the logical extreme, this policy would require a server associated with the ultimate root of a MeSH tree to provide access to any information source classified with any node in that tree. While this policy seems to us unacceptable, the details of an acceptable, systematic alternative remain an open question.

ACKNOWLEDGEMENTS

This work was supported in part by grants LM07089 and LM05513 from the National Library of Medicine, and also by Pittsburgh Supercomputing Center grant number NCR930001P from the NIH National Center for Research Resources.

Reference

[1] Miller PL, Frawley SJ, Powsner SM, Roderer NK. Enfranchising the User for Network Exploration: The Yale Pilot UMLS Information Sources Map. In: Proceedings of the American Medical Informatics Association 1994 Spring Congress. Bethesda, MD: American Medical Informatics Association, p. 126.

[2] Masys DR, Humphreys BL. Structure and Function of the UMLS Information Sources Map. In: Lun KC, Degoulet P, Piemme TE, Reinhoff O, eds. MEDINFO 92. Amsterdam, The Netherlands: North Holland, 1992:1518-21.

[3] Lynch LA. The Client-Server Model in Information Retrieval. In: Dillon M, ed. Interfaces for Information Retrieval and Online Systems: A State of the Art. Westport, CT: Greenwood Press, 1991:301-318.

[4] Garrett JR. Digital Libraries, the Grand Challenges. EDUCOM Review, July-August 1993, pp.17-21.

[5] Obraczka K, et al. Internet Resource Discovery Services. IEEE Computer, Sept. 1993, pp. 8-22.

[6] Bowman CM, Danzig PB, Manber U, Schwartz MF. Scalable Internet Resource Discovery: Research Problems and Approaches. Comm. ACM, 37(8), pp. 98-114.

[7] Metcalfe ES, Frisse ME, Hassan SW, Schnase JL. Academic Networks: Mosaic and World Wide Web. Academic Medicine, 69(4), pp. 270-273.

[8] National Library of Medicine. Medical Subject Headings. Bethesda, MD: NLM, 1994.

[9] Patrick TB, Springer GK, Sista SM, Davison S. Methods for Shared Access to Medical Internet Information Sources. In: Proceedings of the American Medical Informatics Association 1994 Spring Congress. Bethesda, MD: American Medical Informatics Association, p. 123.

[10] Open Software Foundation (OSF). Introduction to OSF/DCE. Englewood Cliffs,NJ: Prentice-Hall, Inc., 1992.

[11] Open Software Foundation (OSF). OSF/DCE Application Development Guide. Englewood Cliffs,NJ: Prentice-Hall, Inc., 1993.

[12] Shirolkar S. Intermediary Servers. MS Thesis, Computer Science Department, University of Missouri-Columbia, 1994.